UNITED STATES PATENT AND TRADEMARK OFFICE

5D

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/898,204	07/03/2001	Todd Poynor	10010393-1	2214

7590 01/05/2005

HEWLETT-PACKARD COMPANY
Intellectual Property Administration
P.O. Box 272400
Fort Collins, CO 80527-2400

EXAMINER	
VU, TUAN A	

ART UNIT	PAPER NUMBER
2124	

DATE MAILED: 01/05/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/898,204	POYNOR, TODD	
	Examiner	Art Unit	
	Tuan A Vu	2124	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *27 September 2004*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-21* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-21* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. This action is responsive to the Applicant's response filed 9/27/2004.

As indicated in Applicant's response, claims 9, 11, and 15 have been amended and claims 16-21 added. Claims 1-21 are pending in the office action.

Claim Rejections - 35 USC § 103

2. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

3. Claims 1-10, 12-14, 16-19, and 21 are rejected under 35 U.S.C. 103(a) as being unpatentable over Cardoza , USPN: 5,630,049 (hereinafter Cardoza), in view of Mahler et al., USPN: 6,675,218 (hereinafter Mahler).

As per claim 1, Cardoza discloses a computer-implemented method for debugging an server operating system kernel (e.g. Fig. 1; *target or server ... system* – col. 4, lines 1-12) executing on a remote processing system that is coupled to a network, the kernel including a debugger control component (e.g. col. 8, lines 42-52; col. 23, lines 8-24; col. 24, line 64 to col. 25, line 14), and the system data processing including a network interface card (*Ethernet card* - col. 14, lines 1-11; col. 10, lines 7-54 – Note: Ethernet card reads on network interface card) and a debugger network component (e.g. device 55 – Fig. 3A), comprising:

detecting debugger messages received over the network (e.g. Fig 3A-B – Note: in a network session based communication scheme, detecting a message is inherent to such established session according to that scheme);

directing the debugger messages to the debugger network component (e.g. col. 9, lines 16-37; Fig. 3A-B – Note: receiving messages and parsing fields of packet is equivalent to directing messages to a debugger network component to filter the packet type);

communicating the debugger messages from the debugger network component to the debugger control component in response to the messages (e.g. col. 8, lines 1-41; Fig. 4-5; col. 27, lines 24-49).

But Cardoza does not explicitly disclose that the network interface card implements a protocol stack, including the layers from physical to application layer. But in view of the protocol layer disclosed (col. 9, lines 2-11) in conjunction with the TCP/IP connection type of network and Ethernet device protocol (e.g. col. 28, lines 23-34; col. 8, line 55 to col. 9, line 45; col. 10, lines 7-54), the network layers as organized in a OSI-like protocol stack is implied. In a method employing user level interface to communicate network messages to a kernel for implementing, testing or debugging communications packet transformation program (see Mahler: *user-level debugging ... code* – col. 5, lines 26-59; col. 10, lines 5-44) analogous to the communicating of debug commands by Cardoza, Mahler discloses a protocol stack for criteria matching and a kernel space/code receiving those debug messages/packets for processing of the validation of those messages (e.g. Fig. 2-4) and a device to intercept and process said packets (e.g. *network interface card* - col. 10, lines 19-29). In case Cardoza's Ethernet card does not already encompass protocol stack executing capability, it would have been obvious for one of ordinary skill in the art at the time the invention was made to implement a kernel logic and network interface card at the server system by Cardoza so

as to include therein a protocol stack processing unit as taught by Mahler's kernel using Mahler's suggested NIC for processing packets (which is also taught by Cardoza) because it is more feasible to perform kernel debugging activities from the user-level thus as to enabling message routing control such that packets implemented from heterogeneous protocol stack can be differentiated, and readdressed or modified/customized and filtered according to the protocol layer and its intended purposes, i.e. allowing user intervention to customize intended use of packets and taking advantage of a kernel filtering capability such as taught Mahler's approach and explained in Mahler's background (see BACKGOUND, col. 1-3).

As per claim 2, Cardoza in combination with Mahler teaches a debugger client system being coupled to the target system but does not explicitly disclose communicating client messages from the debugger control component to the debugger network component, then communicating from there to the protocol stack then from there to the client system. But in view of the communication established by Cardoza to communicate user's bi-directional messages for effecting kernel debug to be conducted to the target system according to the scheme of client originated debugger system (Fig. 1-3; *message to the host computer* - col. 15, lines 4-9), these limitations are implicitly disclosed.

As per claim 3, Cardoza discloses communicating non-debugger messages to a network interface system (e.g. *or a routine ... in the target operating system* - col. 9, lines 11-30; col. 10, lines 39-54 – Note: call back function in libraries matched by message fields identification and device specific information can be routines other than those called for the debugger).

As per claim 4, Cardoza does not explicitly disclose a port number; but official notice is taken that in a socket-based communication involving calls to effect remote procedures, the use of a dedicated port for a particular process operating on such socket was a well-known concept in programming language at the time the invention was made, e.g. telnet port, file transfer port, CGI port. It would have been obvious for one of ordinary skill in the art at the time the invention was made to implement the connection between the host machine and the target machine such that the debugger network component therein is allotted a dedicated port as taught by known concepts because it would make it easy to track the communication link established on the port, thereby enabling logging of data and tracking of errors just from that single port link.

As per claim 5, Cardoza (combined with Mahler) discloses TCP/IP stack (col. 28, lines 23-34).

As per claim 6, Cardoza discloses saving debugger messages in memory of server processing system (e.g. col. 21, lines 10-19).

As per claim 7, Cardoza does not explicitly teach storing client messages from the debugger control component to memory of the server. However, the concept of building a message with a header for transmission across the internet (re claim 5 for Cardoza's TCP/IP communication protocol) inherently requires a temporary storage of the message prior to have it package according to the internet medium and its protocol format; hence the limitation to store client message in the server memory for reformatting before transmission back to the host user machine is implicitly disclosed.

As per claim 8, this claim is the apparatus claim of the method claim 1 and incorporates means for performing the same step limitations(i.e. detecting ; directing;

communicating; performing) as recited therein; hence is rejected using the same corresponding rejections as set forth therein, respectively.

As per claim 9, Cardoza discloses a computing arrangement for debugging an operating system kernel in a server system that is coupled to a client system via a network, comprising:

a memory configured in the target or server system (Fig. 1; *target or server ... system* – col. 4, lines 1-12);

a processor coupled to the memory to execute the O.S. kernel, the kernel including a debugger control component or DCC (e.g. col. 8, lines 42-52; col. 23, lines 8-24; col. 24, line 64 to col. 25, line 14) and a networking subsystem component or NSC (e.g. device 55 – Fig. 3A);

the DCC configured to perform debugging operations in response to debugger messages received (e.g. col. 8, lines 1-19; Fig. 4; col. 27, lines 24-49), and the NSC to provide non-debugging messages to the kernel (col. 9, lines 11-30; col. 10, lines 39-54);

a network interface circuit (NIC; col. 14, lines 1-11; col. 10, lines 7-54) arrangement coupled to the processor and a memory (Fig. 2), the NIC arrangement configured with the network protocol (col. 8, line 55 to col. 9, line 45) and a debugger network component (DNC), the debugger network component configured to communicate debugger messages to the debugger control component in the kernel (e.g. col. 9, lines 16-37; Fig. 3A-B; Fig. 4-5 – Note: device driver and handling of message for loading debug executables read on debugger control component in the kernel debugging system).

But Cardoza does not disclose that the network protocol used by the NIC or Ethernet card arrangement comprises a protocol stack to detect debugger messages and direct messages to the DNC. But this limitation has been addressed in claim 1 using Mahler.

As per claim 10, refer to claim 2 for corresponding rejection.

As per claims 12-13, refer to claims 4-5, respectively.

As per claim 14, based on the teachings by Cardoza's (combined with Mahler's) on using a TCP/IP stack to process messages from the host and back to the host, this claim is rejected using the same rationale as set forth in claim 2.

As per claim 16, Cardoza discloses a method for debugging an operating system kernel, comprising:

executing the operating system on a server data processing system that is coupled to a network (e.g. Fig. 1; *target or server ... system* – col. 4, lines 1-12), wherein the kernel includes a debugger control component (e.g. col. 8, lines 42-52; col. 23, lines 8-24; col. 24, line 64 to col. 25, line 14) and a network interface subsystem – NIC - (e.g. col. 14, lines 1-11; col. 10, lines 7-54);

identifying in a network interface card, debugger messages received over the network, and transmitting debugger messages from the NIC to the debugger network component on the network interface card (col. 9, lines 16-37; Fig. 3A-B);

transmitting the debugger messages from the debugger network component to the debugger control component in the kernel; and performing debugging operations via the debugger control component in response to the debugger messages. (col. 8, lines 1-41; Fig. 4; col. 27, lines 24-49).

But Cardoza does not explicitly discloses identifying debugger messages and non-debugger messages in a protocol stack of the NIC; nor does Cardoza disclose wherein the network interface card implements a protocol stack that includes layers from a physical layer through an application layer and a debugger network component coupled to the protocol stack; and transmitting non-debugger messages from the protocol stack to the network interface subsystem of the kernel. The use of Ethernet card already implies a protocol stack being implemented; but in case Cardoza does not already include such feature in the Ethernet card, this limitation would have been obvious as follows.

According to the rationale in claim 1 using Mahler, a device for intercepting heterogeneously protocol-implemented packets in conjunction with a process for filtering user-customized messages so to effect identifying a specific protocol as well as testing the validity of said messages according to protocol stack by means of a kernel code, the limitation as to provide a protocol stack to a interface device for filtering nature of packets (e.g. *Ethernet driver, divert packets* -col. 5, lines 15-59; col. 10, lines 5-44; *network interface card* - col. 10, lines 19-29) in conjunction kernel debug and support would have been obvious for the same reasons as set forth in claim 1. Hence, the destination/purpose of packets as to whether they are for debugging or for non-debugging so as to be determined by the kernel filtering code by Mahler would have rendered the steps of identifying and transmitting of non-debugging messages --being processed by an interface card among incoming messages from the user— also obvious because of the very concept to use user intervention by Mahler to enable the kernel to re-route or reject appropriate packets according to their intended protocol or application (e.g. debug or non-debug subsystem) following the same rationale as set forth in claim 1.

As per claim 17, Cardoza in combination with Mahler teaches a debugger client system being coupled to the target system but does not explicitly disclose communicating client messages from the debugger control component to the debugger network component, then communicating from there to the protocol stack then from there to the client system. But in view of the communication established by Cardoza to communicate user's bi-directional messages for effecting kernel debug to be conducted to the target system according to the scheme of client originated debugger system (Fig. 1-3; *message to the host computer* - col. 15, lines 4-9), these limitations are implicitly disclosed.

As per claims 18-19, refer to claims 4-5, respectively

As per claim 21, this is a apparatus version of claim 16, and includes means to perform all the corresponding limitations of claim 16; hence is rejected herein with the corresponding rejections as set forth therein.

4. Claims 11, 15 and 20 are rejected under 35 U.S.C. 103(a) as being unpatentable over Cardoza , USPN: 5,630,049, in view of Mahler et al., USPN: 6,675,218, as applied to claims 10, 14 and 16, further in view of Barrett et al., USPN: 5,935,262 (hereinafter Barrett).

As per claim 11, Cardoza discloses writing messages to memory (re claim 6) but does not teach a first shared memory interface coupled to the memory and to the debugger control component; and second shared memory interface coupled to the memory and the debugger control component, both interfaces configured to write debugger and client messages to a shared memory area. The use of shared memory in a paradigm wherein a host client system is interacting using communications via a network or bus interface arrangement or circuitry with a target device for remote debugging

thereof was a known concept in the art of debugging kernel and hardware/software emulation. Cardoza, further discloses a R/W memory for caching network data (Fig. 2) while also suggesting a need for allowing data to be maintained and thus communicated from a common area in the context of debug switching mode (*global data area* - col. 14, lines 1-11). Further, Barrett, in a method to collect debug information from a target network device via an interface card, discloses a shared memory within the interface itself (e.g. col. 25, line 47 to col. 26, line 10; col. 30, lines 30-54; Fig. 23), hence implicitly disclose shared R/W interface. Based on the interrelated communication between the interface card and the debugging kernel as intended by Cardoza in light of Mahler, it would have been obvious for one of ordinary skill in the art at the time the invention was made to provide a shared memory accessible from within the interface arrangement as taught by Barrett so that it can provide writable interfaces to both the DCC and the DNC as disclosed by Cardoza's R/W interface from above. The motivation would be that this would provide a place for information to be commonly available, whether in the target device or in the remote debug commanding device whenever resources allow, for bi-directional access (as in share storage) for debug analysis from the standpoint of the target being debugged (kernel control side or server control component as claimed) and debug control component (network control side) as taught by Barrett; as well as enhancing the bi-directional debug paradigm by which messages or program data as suggested by Cardoza can be accessed or stored or debug mode switching which would enable data to be maintained (via R/W) and re-accessed prior to and after a debug mode has switched.

As per claim 15, this claim corresponds to claim 11; hence is rejected using the rationale as set forth therein.

As per claim 20, Cardoza in combination with Mahler does not disclose writing the debugger messages from the debugger network component to memory of the server data processing system by a first shared memory interface in the network interface card; and writing the client messages from the debugger control component to memory of the server data processing system by a second shared memory interface that is coupled to the debugger control component and executes on the server data processing system. But according to the rejections in claims 6-7, and the teachings by Cardoza to provide network R/W cache as well as a common data area to communicate data from debug mode, in light of shared memory teachings by Barrett, the above limitations would have been obvious based on the rationale as set forth in claim 11.

Response to Arguments

5. Applicant's arguments filed 9/27/2004 have been fully considered but they are not persuasive. Following are the Examiner's observations in regard thereto.

(A) In regard to claim 1, Applicants have submitted that Cardoza does not execute the messages on the NIC nor does Cardoza teach a debugger network component therein (Appl. Rmrks, pg. 7, second to last para). The claim does not recite unequivocal language that would dictate that the NIC is necessarily the unit that performs the execution of messages or directing messages for debug operations; nor does the claim enforce that it is the NIC that contains this debugger network component. From the claim language, i.e. '...that implements a protocol stack, including layers from a physical layer through an application layer, and a debugger network component, comprising: ', it is hard to

determine whether or not this network component is strictly inside the NIC when the NIC implements a protocol stack, notably when the notion that a protocol stack including layers reasonably overwhelms the possibility that this same stack would include this network component. Cardoza discloses an Ethernet interface with an internet card to support message communications and analysis of protocol related data according to portions cited; and as broadly interpreted, the claim limitation concerning the network component being 'included' in the method of debugging has been met by Cardoza.

(B) Applicants have submitted that Mahler's protocol functions implemented on the kernel or user space are not same as network interface card. The current rejection has shown that Mahler provides user-defined program instructions/commands to be tested via protocol-bound messages by a kernel for debug/test environment and also teaches interception of messages by an interface card. The methodology as to use protocol stack inside a validation scheme using kernel code as in test/debug of protocol related packets or messages was a known concept in the art of network interfacing using a NIC. The motivation as set forth in the rejection now presents the point that Mahler is trying to resolve: enabling disparate protocol-originated messages or packets to be validated for their intentions and using the kernel to provide such debug or testing facilities would enable that purpose. And the arguments concerning improper motivation to combine Mahler to Cardoza (Appl. Rmrks, pg. 8, middle) is becoming moot.

(C) Applicants have submitted that claim 2 includes 'limitations of communicating ... Office Action alleges that ... implicitly teaches ... no evidence cited ... how messages to a host computer ... imply a debugger network component ... debugger control component in a kernel' (Appl. Rmrks, pg. 8, bottom). The teachings by Cardoza as cited

relate to a Ethernet card being used in the debug session using a user directed means or unit for deciphering and analyzing commands using protocol libraries/messaging specified in predefined format. The NDP information along with special format have been construed along with the analyzing tool above as network component in conjunction with the network interface card and the Ethernet protocol. The target environment or driver device effecting the debug commands deciphering or executing the debug calls is construed as part of or the debug control component. The necessary movement of data between the control component and the network component have to happen lest there would be no message being analyzed nor would there be any debug command being communicated to the target environment for examining the message content being transmitted; and that is the ground of rejecting claim 2. Further, the claim does not recite clear language as to enforcing that the network component is included in the NIC. As interpreted, the claim amounts to a NIC implementing a protocol stack, a feature made obvious by virtue of Mahler, a network component as described above; and a control component at the receiving end of the debug messages being communicated over via the NIC and the debug network component analysis as shown in Fig. 2-4 of Cardoza.

(D) Applicants have submitted that the alleged obviousness of claim 4 concerning a debug port is construed on hindsight (Appl. Rmrks, pg. 9, top). Since this is an official notice teaching that the reservation of a port for debug in interconnected computers was a known concept, there is no requirement that evidence be provided because the unilateral providing by an Office Action of a reference to corroborate an official notice would negate the use of such notice. The rejection has set forth the reason why combining the known concept with debug interface by Cardoza would be beneficial. On the one hand,

Applicants also admit that debug port was a common concept; on the other, Applicants nevertheless fail to show why such well-known concept being combined as in Cardoza-Mahler teachings would have been improper; hence the hindsight argument amounts to allegation.

(E) As for argument on claim 6 rebutting the Office Action for not providing 'writing the debugger messages to the server memory' by the NIC (Appl. Rmrks, pg. 9, 2nd para), the claim only recites 'further comprising writing ... messages from the ... network component to the memory of the server ... system'. As such, the interpretation is that there would be no NIC writing any messages because there are no specifics as to which entity is being responsible for the act of writing. Thus, the messages have been construed as 'messages from the debugger network component', something proved to being disclosed in Cardoza as cited in the rejection.

(F) Applicants have submitted that the limitations of claims 11 and 15 are not suggested by Cardoza-Mahler-Barrett and that Barrett's shared memory is in the expansion device, not in the server (Appl. Rmrks, pg. 9, bottom 2 paras) and that there is no proper motivation for having a shared memory as alleged from the Office Action (Appl. Rmrks, pg. 10). It is noted that the rejection has now addressed how there is desirability in Cardoza for a shared area for communicating data being manipulated between two debug modes; and in conjunction with the notion of having shared memory in a debug paradigm as by Barrett, a network R/W cache in the network interface by Cardoza, the combination as to provide such R/W interface so that a shared memory can store said manipulated data during debug switch has been set forth for it would enhance the bi-directional aspect of debug by Cardoza using the network component in support of

the control component. Applicants' argument would become moot in light of the re-adjusted reason for combining. Since the notion of memory has been detected as indefinite during the prior Office Action, the amendment to the claims effected herein in this amendment - namely claims 9, 11 - necessitates new reasons to recombine the teachings for addressing changes that would otherwise did not exist when the previous Office Action was constructed. For the sake of argument, the methodology as stakes in the rationale of claim 11 rejection revolves around the use of an interface so that shared data (which inherently entails a shared memory) can be accessed for read and write, the combination to provide such shared storage between the debug target and the debug commanding environment is at the crux in the motivation to combine. Whether this shared memory is implemented in some interface or in the target device (or server system) or in the debug controlling device (or user machine) would not but a minor issue. This is because when resources allow that the shared memory can be implemented in the former or in the latter environment, then that would where such shared memory would be implemented as an obvious feature. The commonality and purposefulness of data being shared is that it provides a R/W interface enabling the access of data as mentioned in the rejection, such being a main obvious feature triggering the motivation to combine, whereas where it is implemented becoming a secondary feature of obviousness.

For the above reasons, the rejection will stand as set forth in the above Action.

Conclusion

Applicant's amendment necessitated the new ground(s) of rejection presented in this Office action. Accordingly, **THIS ACTION IS MADE FINAL**. See MPEP

§ 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the date of this final action.

1. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Tuan A Vu whose telephone number is (272) 272-3735. The examiner can normally be reached on 8AM-4:30PM/Mon-Fri.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Kakali Chaki can be reached on (571)272-3719.

The fax phone number for the organization where this application or proceeding is assigned is (571) 273-3735 (for non-official correspondence – please consult Examiner before using) or 703-872-9306 (for official correspondence) or redirected to customer service at 571-272-3609.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

VAT
December 28, 2004

TODD INGBERG
PRIMARY EXAMINER